Exploiting the Cell BE Architecture on Roadrunner for Large-scale Molecular Dynamics

Sriram Swaminarayan, CCS-2; Timothy C. Germann, T-12; Kai Kadau, T-14

Roadrunner, LANL's latest supercomputer due to be delivered at the end of June 2008, is an IBM Cell Broadband Engine (Cell BE) accelerated Opteron cluster consisting of about 7,000 nodes. Each compute node of the machine has two dual-core Opterons and two dual Cell BE blades. The machine is capable of delivering a peak of 1.4 petaflops double precision, almost all of which (>96%) is delivered by the Cell BEs. Consequently, to achieve good performance, any code that is written *must* utilize the Cell BEs. Complicating this simple axiom are the following:

- The Cell BEs are true local accelerators in that they are connected to the host via PCI Express slots and have no network access. Consequently all network communication (e.g., MPI calls) must be routed through the Opterons on the node.
- The endianness of the Cell BE is different from that of the Opteron, and so any data that is transferred between the Opteron and the Cell BE has to be byte-swapped before the other host can interpret it.
- The Cell BE is a vector machine. To exploit its impressive performance one has to vectorize the kernels of code one plans to run on it.
- Although the Cell BE has 8 GB of main memory, the vector units (called synergistic processing units, or SPUs) have only 256 kB each, called the Local Store, and the user has to explicitly manage the data residing in this local memory.

In this paper we briefly describe our strategy for overcoming each of the above points to port SPaSM, a large-scale massively parallel molecular dynamics (MD) code developed at LANL, to the Roadrunner machine.

The first step in accelerating SPaSM was to identify the computational hot spots. In SPaSM, as in most MD codes, over 95 percent of the time is spent in the force computation, making the selection of the hot spot easy. To compute forces, SPaSM decomposes space into subdomains and computes interactions between atoms in neighboring subdomains. Thus, once we receive boundary subdomains from

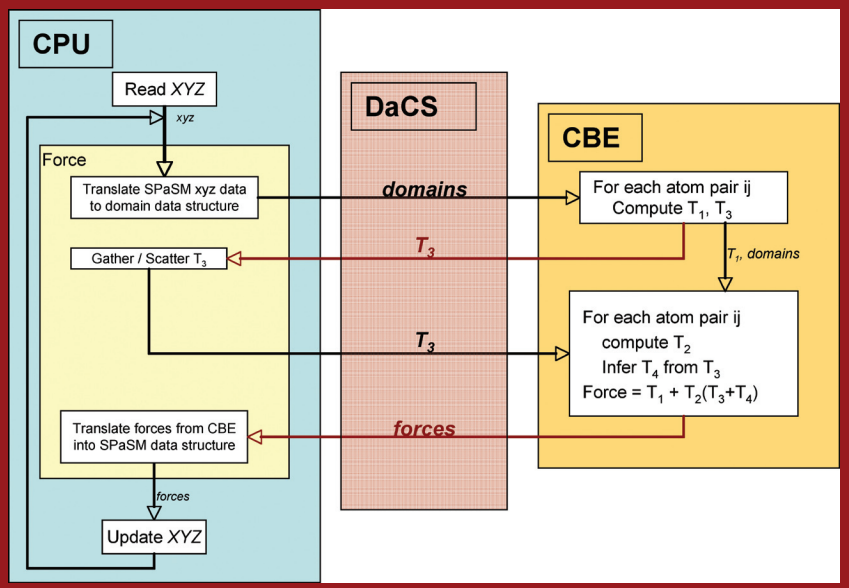

Fig. 1. Schematic of data/control flow for the hybrid implementation of SPaSM on Roadrunner. The box shows the terms in the interaction using the EAM potential.

neighboring processors, the computation is entirely local. Our model therefore was to have the Opterons manipulate the positions of the particles while the Cell BEs computed the forces. This eliminates the need for the Cell BEs to do any communication with any other Cell BE. This model is schematically represented in Fig. 1, which shows the flow of control and data during the course of a simulation. The data that is sent to the Cell BE is endian, converted on the Opteron before transmission, and the forces returned are endian, converted on the Cell BE before transmission. We found the time spent in endian conversion to be an insignificant fraction of the time spent computing forces.

On the Cell BE itself, the data for the atoms (about 200 MB for a million atoms) is resident in main memory. To compute forces, the atoms are *streamed* through the Cell BE one domain at a time. This allowed us to stay within the 256 kB Local Store while still being able to compute forces on all atoms. For our code the Local Store

on the SPU is approximately utilized as follows: 30 kB for the force code, 70 kB for the embedded atom method (EAM) potential tables (three tables of 3000 doubles each), and the rest available for use as a cache for atomic positions and forces.

As a first cut we have implemented a hybrid parallel version of this model using full neighbor lists (i.e., we do not exploit Newton's third law to reduce the computation by a factor of two). In spite of this, for system sizes that are larger than 8,000 atoms per Opteron core (we've tested sizes up to a few million atoms per Opteron core), we get a speedup of about three times over the Opteron version. This is particularly encouraging given that not only are we doing twice the work of the Opteron version, we are achieving this speedup on cells that are about 1/8th the speed of that which will be provided with Roadrunner.

In conclusion, we have ported an existing large-scale MD code running a nontrivial potential to the Roadrunner architecture and achieved good performance gains. By the time Roadrunner is delivered, we expect to improve the performance further, not only because the Cell BEs on Roadrunner are eight times faster, but also because we are redesigning the streaming algorithm to use half-neighbor lists. Thus, although at first blush the Roadrunner architecture appears to be difficult to program, we have demonstrated that it is relatively simple to accelerate existing codes and achieve high levels of performance.

For more information contact Sriram Swaminarayan at sriram@lanl.gov.

Funding Acknowledgments

- Department of Energy, National Nuclear Security Administration, Advanced Simulation and Computing Program